WHAT IS CLAIMED IS:

1.   A method of maintaining a cache memory in a computer system having a first
processor, a first memory, and at least a first cache between the first processor and the
first memory, the method comprising:

performing a first memory access to the first memory by the first processor;

storing a first cache line from the first memory into the first cache;

performing an instruction that enables a cache-invalidate function to be
performed by the first processor upon execution of a resource-synchronization
instruction;

performing the resource-synchronization instruction, wherein at least the first
cache line is invalidated in the first cache;

performing a second memory access to the first memory by the first processor;

storing a second cache line from the first memory into the first cache;

performing an instruction that disables the cache-invalidate function from being
performed by the first processor upon execution of the resource-synchronization
instruction;

performing the resource-synchronization instruction, wherein at least the second
cache line is not invalidated in the first cache.

2.   The method of claim 1, wherein the computer system further includes a second
processor and at least a second cache between the second processor and the first
memory, the method further comprising:

performing a third memory access to the first memory by the second processor;

storing a third cache line from the first memory into the second cache;

performing the instruction that enables the cache-invalidate function to be
performed by the second processor upon execution of the resource-synchronization
instruction;

performing the resource-synchronization instruction, wherein at least the third cache line is invalidated in the second cache, and wherein no data in the first cache is invalidated;

performing a fourth memory access to the first memory by the second processor;

storing a fourth cache line from the first memory into the second cache;

performing the instruction that disables the cache-invalidate function from being performed by the second processor upon execution of the resource-synchronization instruction;

performing the resource-synchronization instruction, wherein at least the fourth cache line is not invalidated in the first cache.

3.   The method of claim 2, wherein the resource-synchronization instruction is a test-and-set instruction.

4.   The method of claim 3, wherein the instruction that enables the cache-invalidate function is an enable-test-and-set-invalidate instruction, and the instruction that disables the cache-invalidate function is an disable-test-and-set-invalidate instruction.

5.   The method of claim 1, wherein the resource-synchronization instruction is a test-and-set instruction.

6.   The method of claim 5, wherein the instruction that enables the cache-invalidate function is an enable-test-and-set-invalidate instruction, and the instruction that disables the cache-invalidate function is an disable-test-and-set-invalidate instruction.

7.   The method of claim 1, wherein by the performing the resource-synchronization instruction, the entire first cache is invalidated.

8.   A computer instruction set comprising:

a resource-synchronization instruction;

an instruction that enables a cache-invalidate function to be performed upon execution of the resource-synchronization instruction; and

an instruction that disables the cache-invalidate function from being performed upon execution of the resource-synchronization instruction.

9.   The computer instruction set of claim 8, wherein the resource-synchronization instruction is a test-and-set instruction.

10.   The computer instruction set of claim 9, wherein the instruction that enables the cache-invalidate function is an enable-test-and-set-invalidate instruction, and the instruction that disables the cache-invalidate function is an disable-test-and-set-invalidate instruction.

11.   The computer instruction set of claim 8, wherein the instruction that enables the cache-invalidate function is an enable-resource-synchronization-instruction-invalidate instruction, and the instruction that disables the cache-invalidate function is an disable-resource-synchronization-instruction-invalidate instruction.

12.   An information-processing system comprising:

a first processor;

a first memory;

at least a first cache between the first processor and the first memory, wherein the first cache caches data accessed by the first processor from the first memory, wherein the first processor executes:

a resource-synchronization instruction;

an instruction that enables a cache-invalidate function to be performed upon execution of the resource-synchronization instruction; and

16

an instruction that disables the cache-invalidate function from being performed upon execution of the resource-synchronization instruction.

13.   The system of claim 12, wherein the resource-synchronization instruction is a test-and-set instruction.

14.   The system of claim 13, wherein the instruction that enables the cache-invalidate function is an enable-test-and-set-invalidate instruction, and the instruction that disables the cache-invalidate function is an disable-test-and-set-invalidate instruction.

15.   The system of claim 12, wherein the instruction that enables the cache-invalidate function is an enable-resource-synchronization-instruction-invalidate instruction, and the instruction that disables the cache-invalidate function is an disable-resource-synchronization-instruction-invalidate instruction.

16.   The system of claim 12, further comprising:

a second processor; and

at least a second cache between the second processor and the first memory, wherein the second cache caches data accessed by the second processor from the first memory, wherein the second processor executes:

the resource-synchronization instruction;

the instruction that enables a cache-invalidate function to be performed upon execution of the resource-synchronization instruction; and

the instruction that disables the cache-invalidate function from being performed upon execution of the resource-synchronization instruction.

17.   The system of claim 16, wherein the resource-synchronization instruction is a test-and-set instruction.

18.   The system of claim 17, wherein the instruction that enables the cache-invalidate function is an enable-test-and-set-invalidate instruction, and the instruction that disables the cache-invalidate function is an disable-test-and-set-invalidate instruction.

19.   The system of claim 16, wherein the instruction that enables the cache-invalidate function is an enable-resource-synchronization-instruction-invalidate instruction, and the instruction that disables the cache-invalidate function is an disable-resource-synchronization-instruction-invalidate instruction.

20.   The system of claim 12, wherein the cache-invalidate function invalidates the entire first cache.

21.   An information-handling system comprising:
    a memory;
    a plurality of processing elements (PEs) including a first processing element (PE), wherein each one of the PEs has a cache associated with that PE, including a first cache associated with the first PE, and wherein each one of the PEs is operatively coupled to the memory; and
    means for enabling and disabling a cache-invalidate function from being performed by each respective PE on its respective cache upon execution of a resource-synchronization instruction by that respective PE.

22.   The system of claim 21, wherein the resource-synchronization instruction is a test-and-set instruction, and wherein the instruction that enables the cache-invalidate function is an enable-test-and-set-invalidate instruction, and the instruction that disables the cache-invalidate function is an disable-test-and-set-invalidate instruction.

18